AMENDMENTS TO THE CLAIMS

Please amend the claims as follows.

1. (Currently Amended) A method for instantiating an object, comprising:
   determining an object type of said object;
   reserving a memory block on a memory structure, the size of said memory block being determined according to said object type, and said memory structure being selected according to said object type; and
   creating a reference structure to said object[[.]],
   wherein the object is written in a dynamically typed language.

2. (Original) The method of claim 1, wherein said determining comprises:
   obtaining a keyword; and
   identifying said keyword.

3. (Original) The method of claim 2, further comprising:
   executing a set of constructor statements if said set contains at least one statement.

4. (Original) The method of claim 3, wherein said keyword identifies said object type as a class.

5. (Previously Presented) The method of claim 4, wherein said memory structure is a heap.

6. (Original) The method of claim 3, wherein said keyword identifies said object type as a function.

7. (Original) The method of claim 6, further comprising:
   optionally returning a value to a calling statement;
   deleting said reference structure; and
   freeing said memory block.

8. (Previously Presented) The method of claim 7, wherein said memory structure is a stack.

9. (Currently Amended) A computer program product comprising:

a computer usable medium having computer readable program code embodied therein configured to instantiate an object, said computer program comprising:

computer readable code configured to cause a computer to

determine an object type of said object;

computer readable code configured to cause a computer to

reserve a memory block on a memory structure, the

size of said memory block being determined according to said object type, and said memory structure being selected according to said object type; and

computer readable code configured to cause a computer to

create a reference structure to said object[[.]],

wherein the object is written in a dynamically typed language.

10. (Original) The computer program product of claim 9, wherein said computer readable code configured to cause a computer to determine an object type further comprises:

computer readable code configured to cause a computer to obtain a keyword; and

computer readable code configured to cause a computer to identify a keyword.

11. (Original) The computer program product of claim 10, further comprising:

computer readable code configured to cause a computer to execute a set of constructor statements if said set contains at least one statement.

12. (Original) The computer program product of claim 11, wherein said keyword identifies said object type as a class.

13. (Previously Presented) The computer program product of claim 12, wherein said memory structure is a heap.

14. (Original) The computer program product of claim 11, wherein said keyword identifies said object type as a function.

15. (Original) The computer program product of claim 14, further comprising:

computer readable code configured to cause a computer to optionally return a value to a calling statement;

computer readable code configured to cause a computer to delete said reference structure; and

computer readable code configured to cause a computer to free said memory block.

16. (Previously Presented) The computer program product of claim 15, wherein said memory structure is a stack.

17. (Currently Amended) A system for instantiating an object comprising:

an interpreter configured so as to differentiate object types;

a storage allocation subsystem configured so as to reserve a storage block on a memory device, said allocation subsystem further configured to select the size of said storage block and said memory device according to said object type; and

an access control subsystem, said access control subsystem creating a reference structure for said object[[.]],

wherein the object is written in a dynamically typed language.

18. (Original) The system of claim 17 wherein said interpreter further comprises:

a lexical analyzer; and

a semantic parser, wherein said analyzer is configured so as to pass tokens representing keywords to said parser, and said parser is configured so as to identify said tokens.

19. (Original) The system of claim 18, further comprising:

a statement execution subsystem, said execution subsystem configured so as to automatically execute a set of constructor statements.

20. (Original) The system of claim 19, wherein said keyword identifies said object as a class.

21. (Previously Presented) The system of claim 20, wherein said memory device is a heap.

22. (Original) The system of claim 19, wherein said keyword identifies said object type as a function.

23. (Original) The system of claim 22, wherein said execution subsystem is further configured so as to:

optionally return a value to an object calling subsystem.

24. (Original) The system of claim 23, wherein said storage allocation subsystem is further configured to automatically delete said reference structure and automatically freeing said storage block after said statement execution subsystem completes the execution of said set of constructor statements and completes the optional return of a value to said object calling subsystem.

25. (Original) The system of claim 24, wherein said memory device is the stack.

26. (Currently Amended) An object instantiation component for an operating system, comprising:

an interpreter configured so as to differentiate object types;

a storage allocation subsystem configured so as to reserve a storage block on a memory device, said allocation subsystem further configured to select the size of said storage block and said memory device according to said object type; and

an access control subsystem, said access control subsystem creating a reference structure for said object[[.]],

wherein the object is written in a dynamically typed language.

27. (Original) The component of claim 26 wherein said interpreter further comprises:

a lexical analyzer; and

a semantic parser, wherein said analyzer is configured so as to pass tokens representing keywords to said parser, and said parser is configured so as to identify said tokens.

28. (Original) The component of claim 27, further comprising:

a statement execution subsystem, said execution subsystem configured so as to automatically execute a set of constructor statements.

29. (Original) The component of claim 28, wherein said keyword identifies said object as a class.

30. (Previously Presented) The component of claim 29, wherein said memory device is a heap.

31. (Original) The component of claim 28, wherein said keyword identifies said object type as a function.

32. (Original) The component of claim 31, wherein said execution subsystem is further configured so as to:

optionally return a value to an object calling subsystem.

33. (Previously Presented) The component of claim 32, wherein said storage allocation subsystem is further configured to automatically delete said reference structure and automatically freeing said storage block after said statement execution subsystem completes the execution of said set of constructor statements and completes the optional return of a value to said object calling subsystem.

34. (Previously Presented) The component of claim 33, wherein said memory device is a stack.

35. (Previously Presented) The computer program product of claim 9, wherein the object includes a definition of one or more objects.

36. (Previously Presented) The system of claim 17, wherein the object includes a definition of one or more objects.

37. (Previously Presented) The component of claim 26, wherein the object includes a definition of one or more objects.

38. (Currently Amended) A method for instantiating an object, comprising:

determining an object type of said object;

reserving a memory block on a memory structure, the size of said memory block being determined according to said object type, and said memory structure being selected according to said object type;

creating a reference structure to said object; and

wherein the object includes a definition of one or more objects <u>of the object type, and wherein the object is written in a dynamically typed language</u>.

39. (Previously Presented) The method of claim 38, wherein said determining comprises:

obtaining a keyword; and

identifying said keyword.

40. (Previously Presented) The method of claim 39, further comprising:

executing a set of constructor statements if said set contains at least one statement.

41. (Previously Presented) The method of claim 40, wherein said keyword identifies said object type as a class.

42. (Previously Presented) The method of claim 41, wherein said memory structure is a heap.

43. (Previously Presented) The method of claim 40, wherein said keyword identifies said object type as a function.

44. (Previously Presented) The method of claim 43, further comprising:
optionally returning a value to a calling statement;
deleting said reference structure; and
freeing said memory block.

45. (Previously Presented) The method of claim 44, wherein said memory structure is a stack.